Hi, Robert:

As a software engineer, I've come in contact with an extensive number of products and code libraries that fall under the category of 'inventions'. With regard to which of these are patentable, and which fall under the category of prior art, I'm sure you well realize that the distinction is fine indeed. In order to provide a more cogent comment in this regard, I was wondering if it would be possible to get a link to the criteria that you use in differentiating between the two.

Generally, I tend to think of a patentable software product as the combination of both an original algorithm and one or more reference implementations of the algorithm, which seems to be in keeping with the USPTO guidelines. Where the USPTO seems to be getting into some level of trouble is in determining whether either the algorithm or the implementation is an original work of the patent author, or if it is sufficiently different or original in its design as a derivative work to qualify as a patentable work of its own.

An aspect of software patenting that I believe causes a great measure of concern in the industry is the perception that a patent removes the invention from the realm of public domain. All modern software products consist of collaborative elements, some publicly available, others as purchased components, all of which go together to form the product, which represents the labor and intellectual capital of a software-producing enterprise.

The concern is that if one of the publicly available components becomes patented, that the definition of the patent may be legally manipulated to encroach or infringe upon the definition of the product that the patented component is included in. This would cause at least some measure of disruption in the enterprise's ability to perform its appointed task. This scenario falls into the category of being able to demonstrate prior art, but now it becomes incumbent upon the user of the component to prove that the component was indeed previously available, either in the public domain or as an original work of the enterprise.

Further, the presence of patented software processes or components may also be the cause of some level of creative inhibition in the software industry, as the time required to search for prior art and patented origins for a particular process or algorithm progressively increases. There is less inclination to be competitive in a particular category of software design, and to cede the field to the patent holder in favor of concentrating energy, effort, and capital into more lucrative channels.

The beneficiaries of this effect are the larger corporations, which have the intellectual and financial resources to monopolize whole segments of software production. This certainly does not benefit the personal inventor, who in many cases is required to release interest in the patent that he or she originated to the employing organization, in return for some small measure of compensation that in many cases does not reflect the utility or contribution of the patented process or algorithm. Correct me if I'm wrong, but wasn't it the original charter of the USPTO to protect the personal inventor from this type of infringement?

I hope that the points that I've brought up make sense in the context of the work that you're doing, and I look forward to discussing these issues further with you.

Sincerely,

Roby Gamboa
Senior Software Engineer
Iris Financial Engineering and Systems, LLC
456 Montgomery St., Suite 800
San Francisco, Ca. 94104
(415) 835-7853
roby@irisfinancial.com